# Tutorial 11

## Exercise 1

Given the following Python code:

```
import math

class MaxHeap(object):

    def __init__(self, mheap, size):

        self.mheap = mheap
        self.size = 0

    def heapify(self, mheap, currentNode, size):
        last = mheap[currentNode]
        child = currentNode * 2 + 1 # set left child
        while child < size:
            if child + 1 < size and mheap[child] < mheap[child + 1]:
                child += 1
            if last >= mheap[child]:
                break
            mheap[currentNode] = mheap[child] # move child up
            currentNode = child # move down a level
            child = currentNode * 2 + 1 # set left child
        mheap[currentNode] = last

    def deleteMax(self, mheap, size):
        # code to be completed

    def initializemheap(self, mheap, size):
        # max heap initialization
        self.size = size
        for j in range(math.trunc((size - 2) / float(2)), -1, -1):
            self.heapify(mheap, j, size)

    def heapSort(self, a, size):
        # code to be completed

    @staticmethod
    def display(a, size):
        i = 0
        while i < size - 1:
            print(str(a[i]) + ", ", end = '')
            i += 1
        print(a[size - 1])
```

(a) Complete the deleteMax() method of the MaxHeap class such that the deleteMax() method deletes the maximum element of a heap and returns it. If the heap is empty, the deleteMax() method returns a dummy integer -1.

(b) Use the initializemheap() method of the MaxHeap class to heapify the binary tree and its array representation in Figure 1. Show the result in both max heap and its array representation.

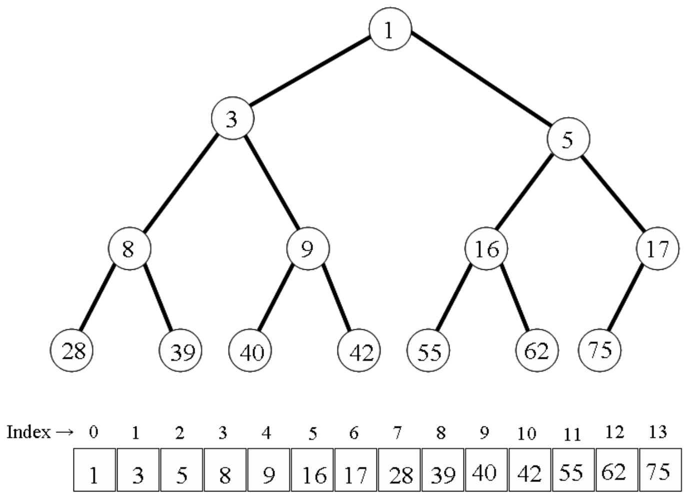

Figure 1

(c) Based on the following HeapSortTest class:

```
class HeapSortTest(object):
    b = [67, 89, 23, 33, 76, 17, 5, 42]
    h = MaxHeap(b, len(b))
    h.size = len(b)
    print("Elements are:")
    h.display(b, len(b))
    h.heapSort(b, len(b))
    print("After sorting by heap sort, the elements are:")
    h.display(b, len(b))
```

complete the heapSort() method of the MaxHeap class such that the heapSort() method sorts the elements b[0 : b.length - 1] using the heap sort method and gives the following output after successfully executing HeapSortTest.

```
Elements are:
67, 89, 23, 33, 76, 17, 5, 42
After sorting by heap sort, the elements are:
5, 17, 23, 33, 42, 67, 76, 89
```

(d) Create a class MaxHeapTest, so that it shows the output after successfully execution**.**

```
Elements in array are:
1, 2, 3, 5, 8, 10, 13, 21, 29, 34, 55, 89, 92, 99
Elements in max heap are:
99, 55, 92, 29, 34, 89, 13, 21, 5, 2, 8, 1, 10, 3
After removing the max from the max heap,
the elements in the max heap are:
92, 55, 89, 29, 34, 10, 13, 21, 5, 2, 8, 1, 3
After sorting by heap sort,
the elements in the max heap are:
1, 2, 3, 5, 8, 10, 13, 21, 29, 34, 55, 89, 92
```